A Study of Independent Component Analysis

Shanti Stewart
School of Electrical Engineering and Computer Science
Oregon State University
stewars2@oregonstate.edu

***Abstract*—Independent Component Analysis (ICA) is a widely used data analysis technique applicable to a broad range of fields including audio signal processing, image processing, biosignal processing, and machine learning. The aim of this paper is to provide a general introduction to the theory and methods of ICA, as well as a popular and efficient ICA algorithm known as FastICA. Additionally, an example of applying FastICA to audio source separation is presented.**

***Index Terms*—independent component analysis, ICA, FastICA, whitening**

I. Introduction

To motivate the topic of independent component analysis (ICA), let us consider a well-known problem in audio signal processing known as the *cocktail party problem*. The setup of the problem is as follows. There is a person, a radio, and two microphones in a single room, and neither the person, radio, nor microphones are moving relative to each other. The person is talking and the radio is playing music simultaneously, and the two microphones are constantly recording. Let us denote the audio source signals of the person (speech) and the radio (music) as $s_1(t)$ and $s_2(t)$, and the recorded audio signals of the two microphones as $x_1(t)$ and $x_2(t)$. This setup is illustrated in Fig. 1.

According to the physics of acoustic theory, sound adds linearly: the audio signal that is recorded in a microphone or perceived in our ear drum is the linear combination of audio signals from multiple sources. In our example, the audio signal recorded at each microphone is a linear combination of the audio signals generated by the person and the radio:

$$x_1(t) = a_1 s_1(t) + b_1 s_2(t) \tag{1}$$

$$x_2(t) = a_2 s_1(t) + b_2 s_2(t) \tag{2}$$

where a_i and b_i are constants that depends on the distance from microphone i to the source. The recorded signals $x_1(t)$ and $x_2(t)$ are called a *linear mixture* of the source signals $s_1(t)$ and $s_2(t)$. For the purpose of this problem, any time delays or other complicating details are ignored.

The goal of this problem is to recover the source signals $s_1(t)$ and $s_2(t)$ just by observing the recorded signals $x_1(t)$ and $x_2(t)$. If the coefficients a_i and b_i are known, then the problem is simply a system of linear equations, which can be solved quite easily. However, if the coefficients a_i and b_i are not known, the problem becomes much more difficult. This problem is an example from a topic called *blind source separation (BSS)*, which encompasses all problems whose goal is to separate mixed sources using only observed data. One approach to solve a subset of BSS problems (namely, those with linear mixture models) is independent component analysis (ICA).

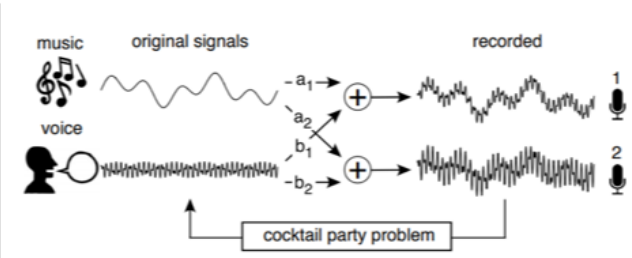

Fig. 1. Illustration of the *cocktail party problem* with a person talking, a radio playing music, and two microphones recording.

II. Problem Formulation

This section explains the framework of ICA and formulates ICA as a general optimization problem.

A. Framework

The basic setup of ICA is the following. Let $\boldsymbol{x}$ be an n-dimensional random vector of the observed data. Each sample of $\boldsymbol{x}$ is assumed to be drawn from an unknown joint distribution $P(\boldsymbol{x})$. Let $\boldsymbol{s}$ be an n-dimensional random vector of n underlying source signals s_i. For the purpose of this section, the number of sources is assumed to be equal to the number of observed signals – this assumption simplifies several steps of the problem formulation.

There are two key assumptions behind the theory of ICA: 1) each source s_i is *statistically independent* from the other sources and 2) the observed data is a *linear mixture* of the sources. Both assumption are reasonable approximations to the underlying physical model in many cases. The linear mixture model can be written as:

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} \tag{3}$$

where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is an unknown square matrix that "mixes" the sources. $\boldsymbol{A}$ is known as the *mixing matrix*. In this section, $\boldsymbol{A}$ is assumed to be nonsingular.

The goal of ICA is to determine an *unmixing* matrix $\boldsymbol{W} \approx \boldsymbol{A^{-1}}$, which is an approximation of the inverse of $\boldsymbol{A}$. The unmixing matrix can then be used to estimate the sources:

$$\hat{\boldsymbol{s}} = \boldsymbol{W}\boldsymbol{x} \tag{4}$$

such that $\hat{\boldsymbol{s}} \approx \boldsymbol{s}$.

B. Initial Strategy

To solve for the unmixing matrix $\boldsymbol{W}$, we first decompose $\boldsymbol{A}$ by singular value decomposition (SVD):

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}} \tag{5}$$

where $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$ is a diagonal matrix of singular values. Since $\boldsymbol{A}$ is nonsingular and therefore full-rank, $\boldsymbol{A}$ has n nonzero singular values and $\boldsymbol{\Sigma}$ consequently is also full-rank and nonsingular. Using the SVD of $\boldsymbol{A}$, $\boldsymbol{W}$ can be written as:

$$\boldsymbol{W} = \boldsymbol{A}^{-\boldsymbol{1}} = (\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}})^{-1}$$

$$\boldsymbol{W} = \boldsymbol{V}\boldsymbol{\Sigma}^{-\boldsymbol{1}}\boldsymbol{U}^{\boldsymbol{T}} \tag{6}$$

where the expression was simplified with the fact that the inverse of an orthogonal matrix is its transpose.

The unmixing matrix $\boldsymbol{W}$ has been decomposed into three pieces, and we can now focus on determining each of these parts separately.

C. Whitening

This subsection explains how to solve for two out of three components of the unmixing matrix $\boldsymbol{W}$: namely, $\boldsymbol{\Sigma}^{-\boldsymbol{1}}$ and $\boldsymbol{U}^{\boldsymbol{T}}$.

1) Covariance of Data: Inspecting the covariance of the data (specifically, the auto-covariance matrix) is a logical place to start, because the covariance contains all (linear) correlations in the data, which is appropriate given that ICA is built upon a linear mixture model. Before computing the covariance, the data is centered:

$$\boldsymbol{x} \leftarrow \boldsymbol{x} - \overline{\boldsymbol{x}} \tag{7}$$

where $\overline{\boldsymbol{x}}$ is the sample mean of $\boldsymbol{x}$.

In order to determine $\boldsymbol{\Sigma}^{-\boldsymbol{1}}$ and $\boldsymbol{U}^{\boldsymbol{T}}$, we have to make an additional assumption that the sources $\boldsymbol{s}$ are *whitened*: $\langle \boldsymbol{s}\boldsymbol{s}^{\boldsymbol{T}} \rangle = \boldsymbol{I}$. Using this assumption, the ICA linear mixture model $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s}$, and the SVD $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}}$, the covariance matrix of $\boldsymbol{x}$ can be expressed as:

$$\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle = \langle (\boldsymbol{A}\boldsymbol{s})(\boldsymbol{A}\boldsymbol{s})^{\boldsymbol{T}} \rangle$$

$$\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle = \langle (\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{s})(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{s})^{\boldsymbol{T}} \rangle$$

$$\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle = \langle (\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{s})(\boldsymbol{s}^{\boldsymbol{T}}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\boldsymbol{T}}) \rangle$$

$$\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}} \langle \boldsymbol{s}\boldsymbol{s}^{\boldsymbol{T}} \rangle \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\boldsymbol{T}} \langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle$$

$$\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^{\boldsymbol{T}}$$

$$\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle = \boldsymbol{U}\boldsymbol{\Sigma}^{\boldsymbol{2}}\boldsymbol{U}^{\boldsymbol{T}} \tag{8}$$

utilizing the fact that the inverse of an orthogonal matrix is its transpose. By using the assumption that the sources are whitened, the covariance of the data can be expressed independently from the sources $\boldsymbol{s}$ and the SVD matrix $\boldsymbol{V}$.

2) Eigendecomposition of the Covariance: In order to solve for $\boldsymbol{\Sigma}^{-\boldsymbol{1}}$ and $\boldsymbol{U}^{\boldsymbol{T}}$, we must take the eigendecomposition of the covariance matrix. Since all covariance matrices (for real-valued vectors) are real-symmetric, $\boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}}$ must admit an eigendecomposition with orthonormal eigenvectors (also known as *orthogonal diagonalization*):

$$\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle = \boldsymbol{E}\boldsymbol{D}\boldsymbol{E}^{\boldsymbol{T}} \tag{9}$$

Since all covariance matrices are positive semi-definite (PSD), all of the eigenvalues of $\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle$ are nonnegative. Furthermore, all covariance matrices of uncorrelated random vectors are positive definite (PD), and consequently all of their eigenvalues are positive. Thus, $\langle \boldsymbol{x}\boldsymbol{x}^{\boldsymbol{T}} \rangle$ has all positive eigenvalues and $\boldsymbol{D}$ is therefore nonsingular.

Considering (8) and (9), both equations show an orthogonal diagonalization of the covariance matrix of the data. Since diagonalizing a real-symmetric matrix with its eigenvectors is unique up to a permutation, the right hand sides of the two equations can be readily compared:

$$\boldsymbol{U} = \boldsymbol{E} \text{ and } \boldsymbol{\Sigma} = \boldsymbol{D}^{\boldsymbol{1/2}} \tag{10}$$

Looking back at (6), the unmixing matrix $\boldsymbol{W}$ can now be written as:

$$\boldsymbol{W} = \boldsymbol{V}\boldsymbol{D}^{-\boldsymbol{1/2}}\boldsymbol{E}^{\boldsymbol{T}} \tag{11}$$

where $\boldsymbol{D}$ and $\boldsymbol{E}$ are the eigenvalues and eigenvectors of the covariance matrix of the data, respectively.

3) Relationship to Whitening: The matrix product $\boldsymbol{D}^{-\boldsymbol{1/2}}\boldsymbol{E}^{\boldsymbol{T}}$ is known as a *whitening filter*, because it *whitens* the data. Whitening is a common technique in signal processing that first rotates the data to align with its orthogonal directions of maximum variance (eigenvectors of the covariance matrix), and then normalizes each axis direction to be of unit variance. Examining (4) and (11), the estimation of the sources can be written as:

$$\hat{\boldsymbol{s}} = \boldsymbol{V}\boldsymbol{D}^{-\boldsymbol{1/2}}\boldsymbol{E}^{\boldsymbol{T}}\boldsymbol{x} = \boldsymbol{V}\boldsymbol{x}_{\boldsymbol{w}} \tag{12}$$

where $\boldsymbol{x}_{\boldsymbol{w}}$ is a whitened version of the (centered) data $\boldsymbol{x}$.

Now the only unknown component of the unmixing matrix $\boldsymbol{W}$ is an orthogonal matrix $\boldsymbol{V}$, which is addressed in the upcoming subsections.

D. Measures of Statistical Independence

To determine the final unknown rotation matrix $\boldsymbol{V}$, we must exploit one of our key assumptions behind ICA: each source s_i is *statistically independent* from the other sources.

1) Definition of Statistical Independence: Statistical independence is the strongest criterion of independence between random variables. It dictates that second-order and all higher-order correlations between the random variables to be zero. In comparison, uncorrelatedness requires only second-order correlations to be zero. Covariance matrices only capture second-order correlations (i.e. linear dependencies) between random variables, and therefore can only be used to manipulate second-order correlations. In the previous subsection, whitening the data $\boldsymbol{x}$ only removes second-order correlations, but

does not remove higher-order correlations. From probability theory, a random vector $\boldsymbol{s} = [\boldsymbol{s_1}, ..., \boldsymbol{s_n}]^T \in \mathbb{R}^n$ is statistically independent if and only if:

$$P(\boldsymbol{s}) = \prod_{i=1}^{n} P(\boldsymbol{s_i}) \tag{13}$$

where $P(\boldsymbol{s})$ is the joint probability density of $\boldsymbol{s}$. In words, $\boldsymbol{s}$ is statistically independent if and only if its joint probability density can be factored as a product of the individual probability densities of its elements $\boldsymbol{s_i}$.

2) Multi-Information and Entropy: In order to minimize the statistical independence of the sources $\boldsymbol{s_i}$, we need some measure of statistical dependence/independence. A natural measure from the branch of mathematics known as information theory is called the multi-information, which is defined as:

$$I(\boldsymbol{s}) = \int P(\boldsymbol{s}) \log_2 \left(\frac{P(\boldsymbol{s})}{\prod_{i=1}^{n} P(\boldsymbol{s_i})} \right) d\boldsymbol{s} \tag{14}$$

The multi-information is a measure of statistical *dependence*, meaning that it has larger values for higher statistical dependencies. The multi-information is always nonnegative and achieves a minimum of zero when $\boldsymbol{s}$ is completely statistically independent. (This is readily apparent by the fact that the argument of the log equals 1 when complete statistical independence exists.)

The final step of ICA can now be expressed concisely: determine a rotation matrix $\boldsymbol{V}$ that minimizes the multi-information of $\hat{\boldsymbol{s}}$, where $\hat{\boldsymbol{s}} = \boldsymbol{V}\boldsymbol{x_w}$ and $\boldsymbol{x_w}$ is a whitened version of the (centered) data $\boldsymbol{x}$. Finding a rotation matrix that singlehandedly removes all higher-order correlations seems like a formidable task, but if the ICA assumptions are correct, then it is achievable.

The multi-information can be written as a function of *entropy*. The entropy of a probability distribution is a measure of its uncertainty and is defined as:

$$H(s_i) = -\int P(s_i) \log_2 \left(P(s_i) \right) ds_i \tag{15}$$

With a little manipulation, the multi-information can be written as:

$$I(\boldsymbol{s}) = \sum_{i=1}^{n} H(s_i) - H(\boldsymbol{s}) \tag{16}$$

To simplify this expression further, let us substitute in $\hat{\boldsymbol{s}} = \boldsymbol{V}\boldsymbol{x_w}$:

$$I(\boldsymbol{s}) = \sum_{i=1}^{n} H([\boldsymbol{V}\boldsymbol{x_w}]_i) - H(\boldsymbol{V}\boldsymbol{x_w})$$

$$I(\boldsymbol{s}) = \sum_{i=1}^{n} H([\boldsymbol{V}\boldsymbol{x_w}]_i) - \left(H(\boldsymbol{x_w}) + \log_2 \left(det(\boldsymbol{V}) \right) \right)$$

$$I(\boldsymbol{s}) = \sum_{i=1}^{n} H([\boldsymbol{V}\boldsymbol{x_w}]_i) - H(\boldsymbol{x_w}) \tag{17}$$

where we have utilized an expression that relates the entropy of a probability distribution that undergoes a linear transformation and the fact that the determinant of the rotation matrix $\boldsymbol{V}$ equals 1.

With (17), we can now write ICA as an optimization problem that finds the optimal rotation matrix $\boldsymbol{V^*}$ that minimizes the multi-information of $\hat{\boldsymbol{s}}$:

$$\boldsymbol{V^*} = \underset{V}{\operatorname{argmin}} \sum_{i=1}^{n} H([\boldsymbol{V}\boldsymbol{x_w}]_i) \tag{18}$$

In practice, computing the entropy of a finite data set (i.e. a sampled probability distribution) is difficult and generally not robust to noise, so various approximations of entropy are used, as will be seen in the next section.

III. FastICA Algorithm

The FastICA algorithm, developed by *A. Hyvarinen* [2] at the Helsinki University of Technology, is a widely used algorithm for solving ICA with good efficiency and convergence properties. As with most ICA algorithms, FastICA searches for an orthogonal rotation matrix ($\boldsymbol{V}$), such that the statistical independence of the projection of the whitened data onto this matrix is maximized. FastICA uses an approximation of negentropy as a measure of statistical independence.

A. Negentropy

Before presenting the FastICA algorithm, we must first introduce the concept of negentropy. Negentropy is defined as:

$$\boldsymbol{J(s)} = H(\boldsymbol{s_{gauss}}) - H(\boldsymbol{s}) \tag{19}$$

where $\boldsymbol{s_{gauss}}$ is a Gaussian random vector of the same covariance matrix of $\boldsymbol{s}$. As can be seen from (19), negentropy is a nonnegative quantity that reaches a minimum of zero if and only if $\boldsymbol{s}$ is Gaussian. Negentropy is a measure of nongaussianity, since it takes on larger values for random vectors that are very different from their Gaussian equivalents (same covariance matrix) and is minimized when the random vector is Gaussian. The FastICA algorithm utilizes the fact that nongaussianity can be seen to be equivalent to statistical independence and uses an approximation of negentropy in its implementation.

B. FastICA for One Unit

The overall FastICA algorithm works by finding the individual independent components separately (with an intermediate modification). The following is the FastICA algorithm for one unit (independent component):

Input: $\boldsymbol{x} \in \mathbb{R}^n$ = observed data
Output: $\boldsymbol{b} \in \mathbb{R}^n$ = independent component of data (weight vector)
$\boldsymbol{b} \leftarrow$ random vector of length n
while $\boldsymbol{b}$ is not converged **do**
  $\boldsymbol{b} \leftarrow \langle \boldsymbol{x} g(\boldsymbol{b^T x}) \rangle - \langle g'(\boldsymbol{b^T x}) \rangle \boldsymbol{b}$
  $\boldsymbol{b} \leftarrow \boldsymbol{b} \ / \ \|\boldsymbol{b}\|$
end while

The function $g()$ and its derivative $g'()$ is a hyperparameter of the algorithm, and can be experimented with to achieve

better performance. *A. Hyvarinen* [2] recommends the set of functions

$$g(u) = tanh(u) \text{ and } g'(u) = 1 - tanh^2(u)$$

for general-purpose applications, and states that the set of functions

$$g(u) = ue^{-u^2/2} \text{ and } g'(u) = (1 - u^2)e^{-u^2/2}$$

are typically more robust.

C. FastICA for Multiple Units

The one-unit algorithm outlined in the previous subsection clearly cannot be used in series to find all of the desired independent components, because the weight vector $\boldsymbol{b}$ would likely converge to the same independent component every time. To prevent different weight vectors from converging to the same value, the projected outputs $\boldsymbol{b_1^T x}, ..., \boldsymbol{b_n^T x}$ must be decorrelated after every iteration. The following algorithm achieves this using a decorrelation method that is very similar to the Gram-Schmidt procedure.

Inputs: $\boldsymbol{x} \in \mathbb{R}^n$ = observed data, c = number of desired independent components
Output: $\boldsymbol{s} \in \mathbb{R}^c$ = estimated sources (whitened)
for k = 1 to c **do**
  $\boldsymbol{b_k} \leftarrow$ random vector of length n
  while $\boldsymbol{b_k}$ is not converged **do**
    $\boldsymbol{b_k} \leftarrow \langle \boldsymbol{x} g(\boldsymbol{b_k^T x}) \rangle - \langle g'(\boldsymbol{b_k^T x}) \rangle \boldsymbol{b_k}$
    $\boldsymbol{b_k} \leftarrow \boldsymbol{b_k} - \sum_{j=1}^{k-1} (\boldsymbol{b_k^T b_j}) \boldsymbol{b_j}$
    $\boldsymbol{b_k} \leftarrow \boldsymbol{b_k} \,/\, \|\boldsymbol{b_k}\|$
  end while
end for
$\boldsymbol{B} \leftarrow [\boldsymbol{b_1}, ..., \boldsymbol{b_c}]$
$\boldsymbol{s} \leftarrow \boldsymbol{B^T x}$

In this algorithm $\boldsymbol{B}$ is equivalent to the transpose of the final rotation matrix $\boldsymbol{V^T}$. Additionally, the output of the algorithm is a whitened version of the sources $\boldsymbol{s}$.

IV. Application: Audio Source Separation

This section details the results of applying the FastICA algorithm to a simple audio source separation example.

A. Data Sets

FastICA was applied to three different sets of audio files [4], each with two recordings (microphones) and two desired sources and a sampling frequency of 16 kHz. The first set of audio files is a recording of two different people simultaneously counting from one to ten (in two different languages), the second set is a recording of one person counting from one to ten while music is playing in the background, and the third set is a recording of two people talking (not counting) simultaneously with some light background noise.

B. Methods

The version of (multiple-unit) FastICA described in the previous section (Section II III B) was used, with contrast functions $g(u) = tanh(u)$ and $g'(u) = 1 - tanh^2(u)$. Additionally, a set number of iterations (set to 100) was used for the inner loop of the algorithm, as opposed to a convergence test, for simplicity of implementation. After applying FastICA to the data, an estimate of the mean and variance was added back in to the estimated source signals. The mean of the original (uncentered) data was close to zero, so adding the mean back in did not make much of a difference. However, it was essential to add the variance back in to hear the estimated source signals (in the .WAV format, the volume of the audio depends on the magnitude of the signal values).

C. Results

Out of the three sets of audio files, the FastICA algorithm seemed to perform the best on the second set (counting and music), so the following results will be for this set of recordings.

We can visually evaluate the results of the algorithm with simple time-domain plots. Fig. 2 shows a plot of the observed audio signals $x_1(t)$ and $x_2(t)$ in the time domain. It can be seen that the counting (from one to ten) is more apparent in the first recording $x_1(t)$, from the roughly periodic spikes in signal value. This reflects the fact that microphone 1 is closer to the person counting, as was mentioned in the introduction section. After applying FastICA to $x_1(t)$ and $x_2(t)$, the estimated source audio signals are displayed in Fig. 3. One can readily see that source $s_1(t)$ contains more information of counting than music, as can be heard in the output audio files. Additionally, the magnitude of $s_1(t)$ is much smaller than the magnitude of $s_2(t)$, which exemplifies one of the shortcomings of ICA.

Another way to visually evaluate the results is to examine the scatter plots of the data and estimated sources (clearly, this is possible only with 2-dimensional data). Looking at Fig. 4, we can see that x_1 and x_2 are fairly strongly correlated; this is expected, because when a sound (from either source) increase or decreases, both microphones will reflect this change in volume somewhat similarly (since neither microphone is extremely close or far away from either source). Examining the scatter plot of the estimated sources in Fig. 5, we can see evidence of the whitening step in ICA since no one direction has significantly more variance than the others.

Since the medium of the data is audio, the best evaluation is to actually listen to the results: the audio output files (in .WAV format) can be found at https://github.com/shantistewart/Independent-Component-Analysis/tree/master/Code[1] As can be heard, $s_1(t)$ contains noticeably more sound of counting than music, although the total volume is much quieter than the other audio files. And as expected, $s_2(t)$ contains more sound of music than counting.

[1]The audio files for this example are named "music_x1.wav", "music_x2.wav", "music_s1.wav", "music_s1.wav", where "x" represents the observed signals and "s" represents the estimated source signals.

Clearly, these results are far from perfect: each estimated source is significantly contaminated with the other. However, this problem of blind source separation is inherently difficult, and the fact that a linear transformation of the data can partly recover the source is quite remarkable.

REFERENCES

[1] A. Hyvärinen and E. Oja, "Independent Component Analysis: Algorithms and Applications," Neural Networks, vol. 13, no. 4-5, pp. 411–430, 2000.

[2] A. Hyvarinen, "Fast and Robust Fixed-Point Algorithms for Independent Component Analysis," IEEE Transactions on Neural Networks, vol. 10, no. 3, pp. 626–634, 1999.

[3] J. Shlens, "A Tutorial on Independent Component Analysis," arxiv, 14-Apr-2014. [Online]. Available: https://arxiv.org/pdf/1404.2986.pdf.

[4] "Blind Source Separation of Recorded Speech and Music Signals," Blind Source Separation: Audio Examples. [Online]. Available: https://cnl.salk.edu/ tewon/Blind/blind_audio.html.

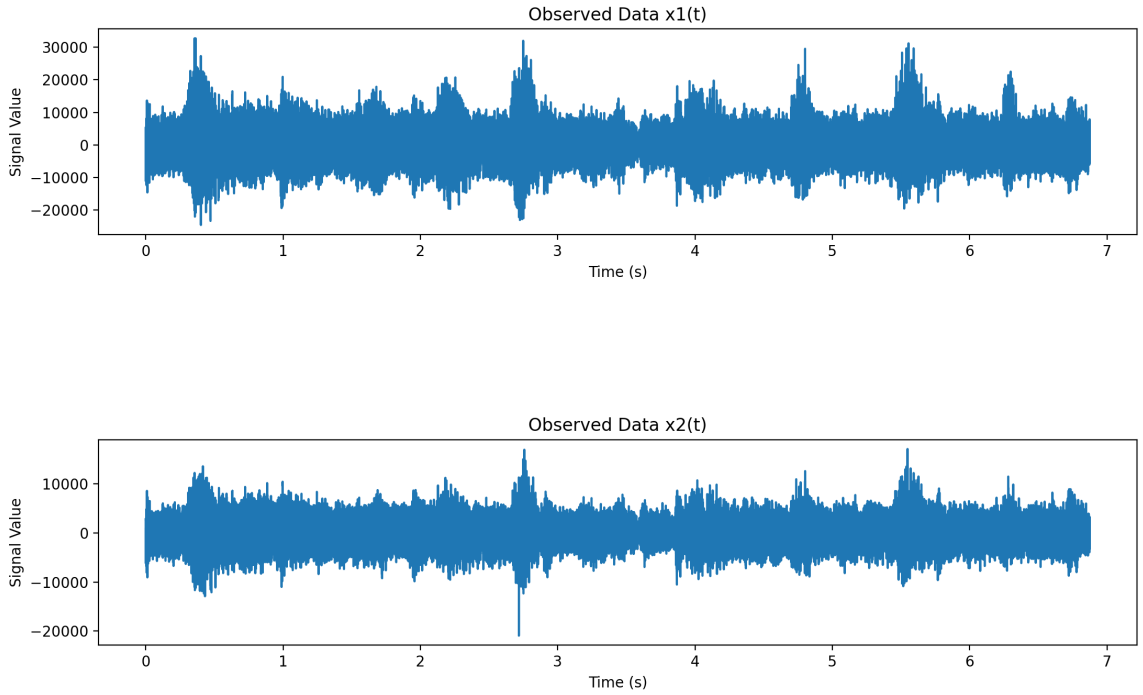

Fig. 2. Plot of the observed audio signals $x_1(t)$ and $x_2(t)$ in the time domain.

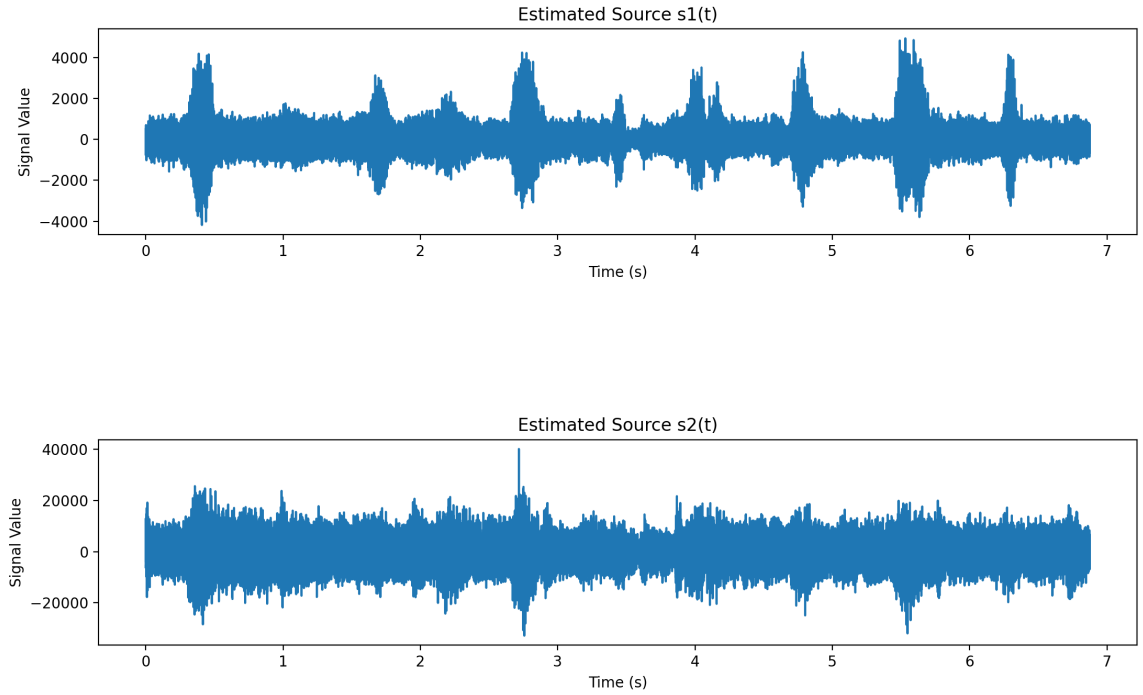

Fig. 3. Plot of the estimated source audio signals $s_1(t)$ and $s_2(t)$ in the time domain.

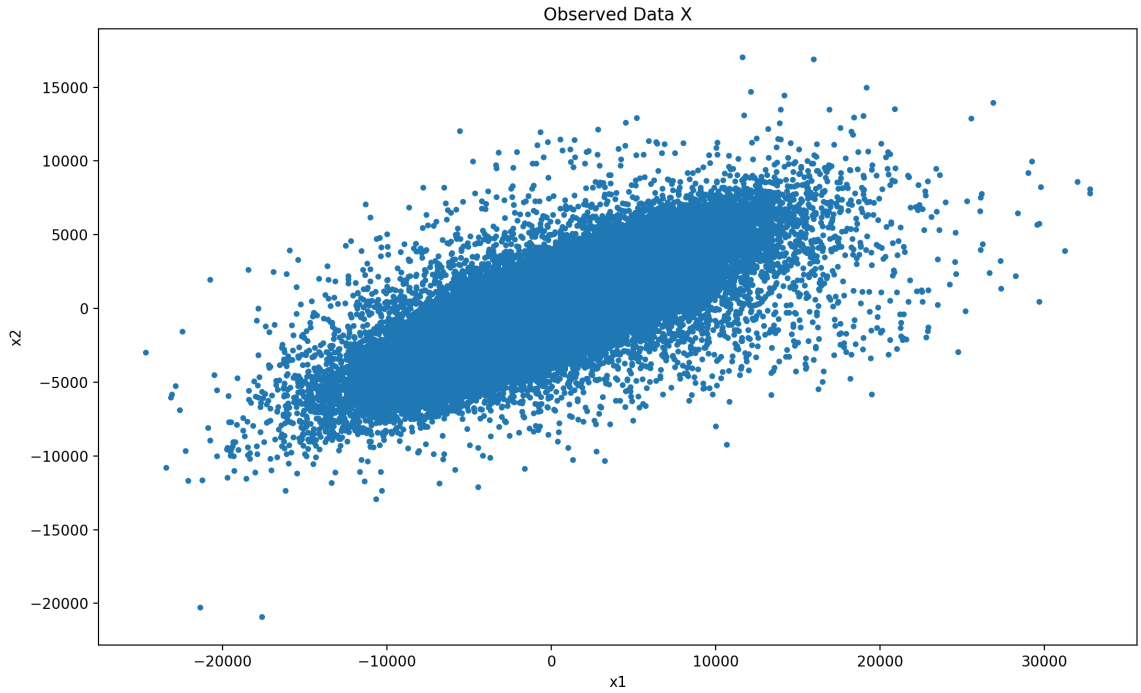

Fig. 4. Scatter plot of the observed audio signals x_1 and x_2.

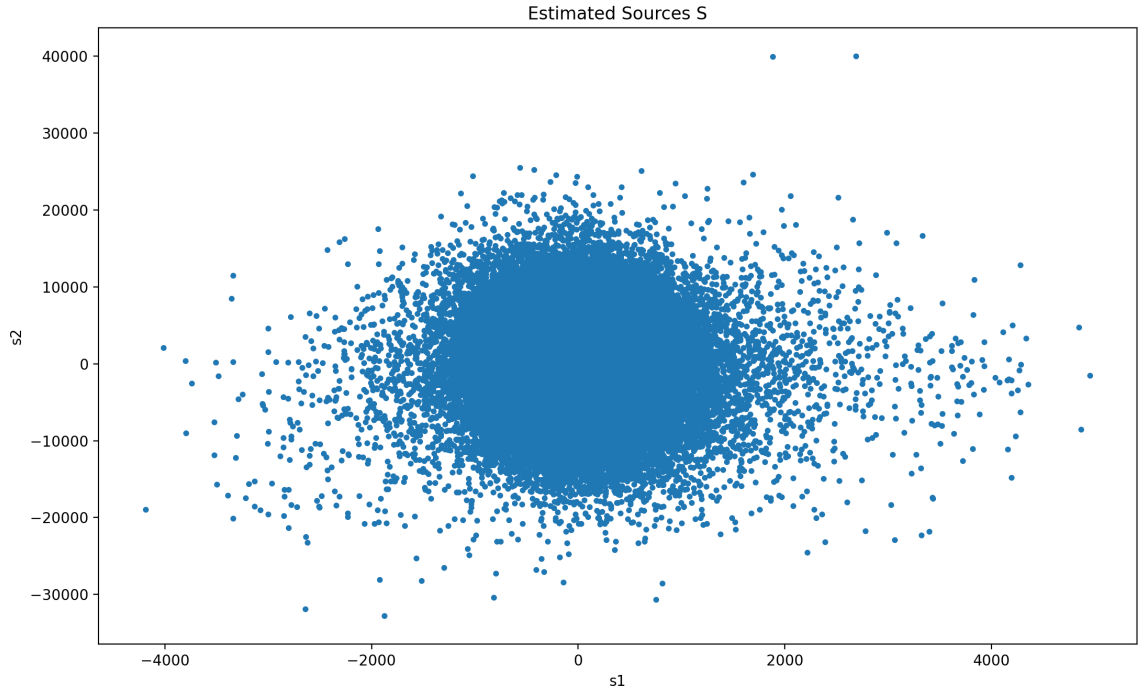

Fig. 5. Scatter plot of the estimated source audio signals s_1 and s_2.